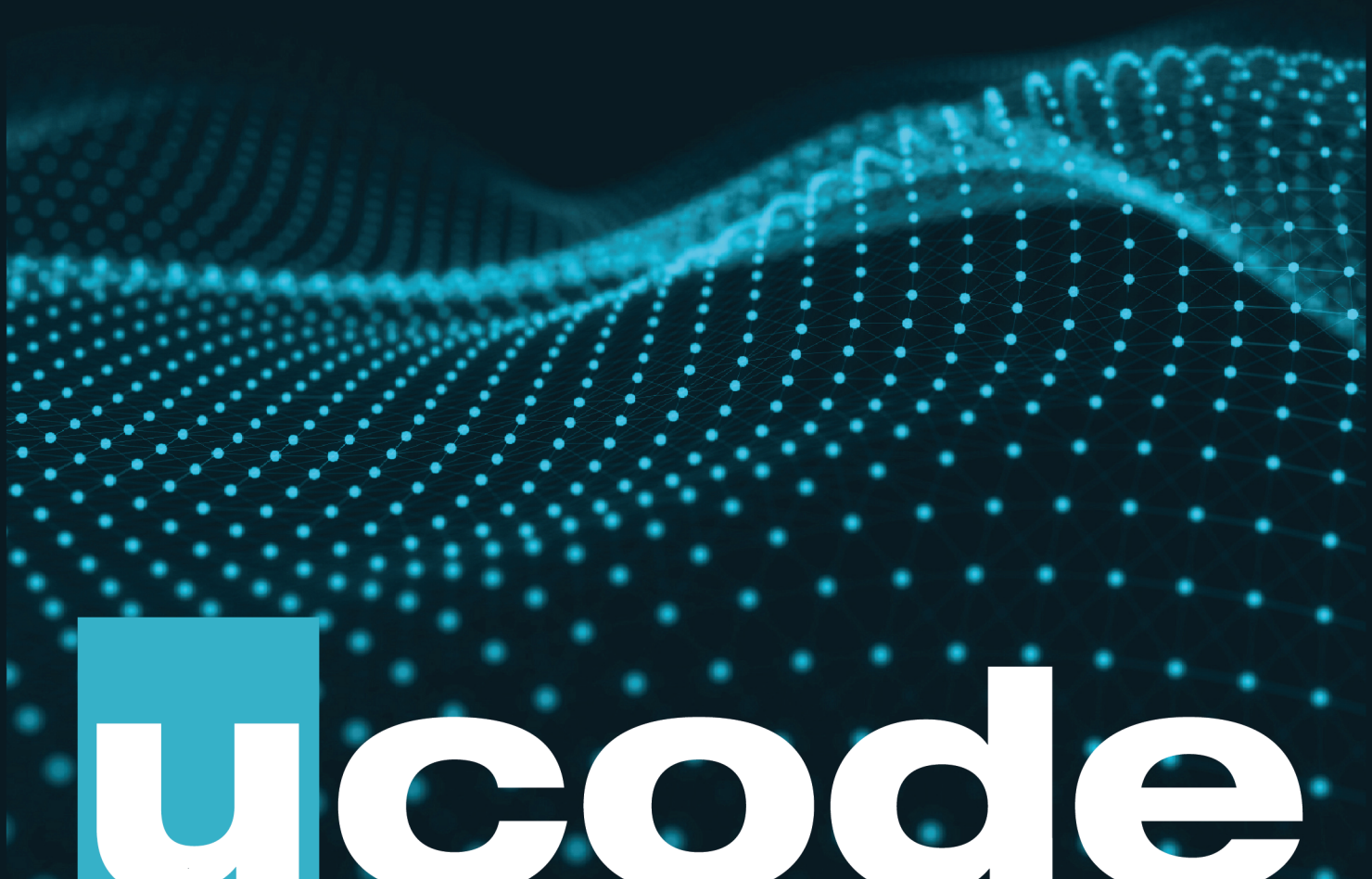
Race 00

Half Marathon C++

August 25, 2020



ucode

Contents

ucode

Engage

DESCRIPTION

Hey, guys!

We are sure that you have gained a lot of useful knowledge and information about the `C++` programming language.

This is your first opportunity to develop a C++ program in a team. Moreover, the main idea of this challenge is game development. You need to familiarize yourself with the basic stages and concepts of game development. C++ is often used in this industry, so it is the right tool for gaining practical experience.

Game development is a software development process, as a video game is software with art, audio, and gameplay. A game can be created both by very few people with a limited budget, or by a large company with hundreds of employees and video game publisher funding. The duration and cost of development depends on the complexity of the project. Planning is important for both individual and group projects. Development of a commercial game usually includes the following stages:

- Pre-production or design phase is a planning phase of the project focused on idea and concept development and production of initial design documents. During this phase game concept and prototype must be created
- Production is the main stage of development when assets and source code for the game are produced. This stage includes design, programming, level creation, art and audio production, etc.
- The release phase may require to meet milestones set by a publisher. Such milestones may be, for example, first playable, alpha or beta game versions

Roles and positions in the development of video games:

- The game designer invents the concept of the game, its main features. Designs basic game mechanics, gameplay, character images, adjusts them throughout development
- The manager coordinates the work of people involved in creating the game
- The programmer writes and refines the code of the game. Creates basic game mechanics, project file architecture
- The artist creates screensavers, sprites, textures, three-dimensional models, interface design. Often there is a distribution where one artist deals with textures, another with special effects, animation, etc.
- The sound engineer performs recording and processing of sounds, music, speech
- The composer writes music. The music itself can be played for recording by an orchestra or a band
- The writer composes the game script, the overall storyline and the individual events. Writes texts of dialogues, descriptions of objects and characters
- The actor is involved in capturing the movements of characters and shooting videos
- The voice actor voices the character's voice, voiceover
- The level designer arranges ready objects, creates story events, sets movement of objects and characters by level

- The PR-manager promotes the game, distributes its advertising, holds presentations
- The tester plays different versions of the game, tests its work in different situations, identifies and documents errors and shortcomings

During this challenge, you must go through all the phases of game development as well as to try yourself in different roles of game production.

So, combine your thoughts, ideas, skills and get started.

BIG IDEA

Game development.

ESSENTIAL QUESTION

How to create a high-end game using the knowledge gained?

CHALLENGE

Create a distinguished `snake-like` game.

Investigate

GUIDING QUESTIONS

We invite you to find answers to the following questions. By researching and answering them, you will gain the knowledge necessary to complete the challenge. To find answers, ask the students around you and search the internet. We encourage you to ask as many questions as possible. Note down your findings and discuss them with your peers.

- What roles exist in game development?
- How to distribute various game development roles in a small team and release the game?
- What graphics libraries exist in C++?
- How to choose the graphics library that meets the challenge's requirements the best?
- What tools exist to create a game in C++?
- What are the gameplay mechanics of a snake-like game?
- How to build a great game UI/UX?
- Which stages of game production do you need to go through to implement a complete solution?
- What limitations in product implementation do you need to deal with?
- What are the elements of a great game design?
- What makes a game addictive?
- How must the gameplay look like?
- How to track user keyboard events?
- How to check the conditions of the game finish?

GUIDING ACTIVITIES

Complete the following activities. Don't forget that you have a limited time to overcome the challenge. Use it wisely. Distribute tasks correctly.

- Meet with your teammate. Discuss how you will organize teamwork. Be supportive and help each other. Remember you are a team and effort of everyone matters. You can search for needed information together and comply your findings.
- Choose a gitflow for effective team interaction.
- Read this story in detail.
- Find and play an old-style snake game.
- Create a work plan for this challenge. Make sure you clearly understand what you need to do.
- Identify and revise the topics that you have struggled with during the Sprints. This challenge is a good opportunity to refresh and gain new knowledge and skills.
- Choose wisely the graphics library that you will use for the game creation (SDL, SFML, OpenGL, etc.).
- Research OOP to discover how to work with classes.

- Clone your git repository that is issued on the challenge page in the LMS.
- Distribute tasks between all team members.
- Start to develop the solution. Offer improvements. Test your code.
- Communicate with students and share information.

ANALYSIS

Analyze your findings. What conclusions have you made after completing guiding questions and activities? In addition to your thoughts and conclusions, here are some more analysis results.

- Challenge has to be carried out by the entire team.
- Each team member must understand the challenge and realization, and be able to reproduce it individually.
- It is your responsibility to assemble the whole team. Phone calls, SMS, messengers are good ways to stay in touch.
- You can proceed to `Act: Creative` only after you have completed all requirements in `Act: Basic`. But before you begin to complete the challenge, pay attention to the program's architecture. Take into account the fact that many features indicated in the `Act: Creative` require special architecture. And in order not to rewrite all the code somewhen, we recommend you initially determine what exactly you will do in the future.
- Be attentive to all statements of the story.
- Analyze all information you have collected during the preparation stages. Try to define the order of your actions.
- Perform only those tasks that are given in this document.
- Submit your files using the layout described in the story. Only useful files allowed, garbage shall not pass!
- Tasks in `shell` must be executed with `zsh`.
- Compile files with commands `cmake . -Bbuild && cmake --build ./build` that will call `CMake` and build an app.
- Pay attention to what is allowed in a certain task. Use of forbidden stuff is considered a cheat and your tasks will be failed.
- Complete tasks according to the rules specified in the Google C++ Style Guide.
  But there are several exceptions for the guide listed below:
  - you can use `#pragma once` directive instead of `#ifndef ... #define`
  - variables can be written in `mixed case`
  - class data members must begin with `m_` prefix (m for member)
  - indent 4 spaces at a time
  - each line of text in your code must be at most 120 characters long
  - ignore the sections Inputs and Outputs and Legal Notice and Author Line in the style guide

- The solution will be checked and graded by students like you. Peer-to-Peer learning.
- You must use the this project structure consisted of logical parts: the main application, own libraries and 3dparty libraries. Each individual unit must know only about its resources and, if necessary, link other libraries to itself.

```
>tree project_dir --dirsfirst --charset=ascii
project_dir
|-- app
|   |-- src
|   |   |-- CMakeLists.txt
|   |   `-- ...
|   |-- resources
|   |   `-- ...
|   |-- CMakeLists.txt
|   `-- main.cpp
|-- lib1
|   |-- CMakeLists.txt
|   `-- ...
...
|-- libN
|   |-- CMakeLists.txt
|   `-- ...
|-- 3dparty
|   |-- lib1
|   |  |-- CMakeLists.txt
|   |  `-- ...
|   ...
|   |-- libN
|   |  |-- CMakeLists.txt
|   |  `-- ...
`-- CMakeLists.txt
```

- If you have any questions or don't understand something, ask other students or just Google it.
- In the name of Talos, use your brain!

Act: Basic

NAME

Chaurus Reaper

DIRECTORY

`./`

BINARY

race00

LEGEND

"Those venturing in Skyrim's deepest underground reaches should be wary of the Chaurus, a giant insect that spits poison and bites with its razor-sharp mandibles."

DESCRIPTION

Create a simple `snake`-like game. Chaurus Reaper will be your `snake`.

The program-game meets the following requirements:

- It consists of at least three screens:
  - `New game`
  - `Leaderboard`
  - `Gameplay`
- The playing area consists of square `blocks` along which Chaurus Reaper can move. Square blocks consists of at least 4 pixels
- The edges of the playing area can't be traversed.
- The size of the playing area is determined by `width` and `height`, passed as command-line arguments
- If there is an incorrect number of arguments or the arguments are incorrect, the program-game prints `usage: ./race00 [width] [height]`
- Chaurus appears in the middle of the game area with length of 4 blocks
- Chaurus moves forward automatically
- Each block of Chaurus's tail follows the head path
- Chaurus is able to turn left or right using the arrows on the keyboard
- The game area has at least one item of food
- The food position is random but the food can't appear in Chaurus's position
- When Chaurus eats one item of food, one block is added to the tail and food disappears
- Speed of Chaurus depends on its length. The longer Chaurus is - the slower it is
- Every 4 seconds the size of Chaurus is reduced by 1 block
- If Chaurus's length is less than 2 blocks - the game is over

- When Chaurus eats one item of food - the timer restarts
- If Chaurus hits an edge of game area or one of its own blocks - the game is over
- After the game is over the `New game` screen appears
- After each game the final score is saved and displayed
- `Leaderboard` screen is a list containing the top 10 results. It saves results between app launch

You are allowed to use ONLY C++ features that you have learned during the Sprints. Inheritance, operator overloading, etc. are forbidden.

Act: Creative

DESCRIPTION

Listed below are a few ideas you can add to your program. But do not focus only on them you can come up with everything you want in order to improve your game.

- The game area has obstacles. If Chaurus hits them - the game is over
- Chaurus has several lives before the game is over
- Multiplayer mode
- Some fancy textures! The game looks pretty, Chaurus's head and tail differs from its body
- The ability to change the look of Chaurus
- Difficulty levels
- Bonus food that gives more points
- Different sounds on movement while eating or hitting an obstacle, etc.
- More interesting and informative game screens
- Other creative features

Share

PUBLISHING

Last but not least, the final stage of your work is to publish it. This allows you to share your challenges, solutions, and reflections with local and global audiences. During this stage, you will discover ways of getting external evaluation and feedback on your work. As a result, you will get the most out of the challenge, and get a better understanding of both your achievements and missteps.

To share your work, you can create:

- a text post, as a summary of your reflection
- charts, infographics or other ways to visualize your information
- a video, either of your work, or a reflection video
- an audio podcast. Record a story about your experience
- a photo report with a small post

Helpful tools:

- Canva - a good way to visualize your data
- QuickTime - an easy way to capture your screen, record video or audio

Examples of ways to share your experience:

- Facebook - create and share a post that will inspire your friends
- YouTube - upload an exciting video
- GitHub - share and describe your solution
- Telegraph - create a post that you can easily share on Telegram
- Instagram - share photos and stories from ucode. Don't forget to tag us :)

Share what you've learned and accomplished with your local community and the world. Use #ucode and #CBLWorld on social media.